A Series-parallel XML Multisignature Scheme for XML Data Authentication

Baolong Liu, Joan Lu, and Jim Yip,

School of Computing & Engineering, University of Huddersfield, Queensgate, Huddersfield, HD1 3DH, UK

Abstract
XML signature technology is the major approach for ensuring XML data authentication. An XML signature should satisfy multiple data authentication requirements for XML data which pass a hierarchical network of responsibilities. Through investigation, existing XML multisignature schemes can not satisfy this requirement. This paper presents a series-parallel XML multisignature scheme based on Lu's XML multisignature scheme. In the scheme presented, signers are divided into series or parallel subgroups according to their relationship, making the multisignature process closer to a natural signature-generation process. The scheme builds an XML data integrity-checking pool to provide integrity-checking for decomposed XML data. With this integrity-checking pool, signers can check integrity without the cooperation of others checkers. Testing shows that the scheme presented has a higher efficiency than repeated DSA or RSA, and satisfies application requirements in practice.

Key words:
XML security, XML data authentication, XML multisignature, DSA, RSA

1. Introduction

The wide spread of XML applications has presented significant need for XML security. XML data authentication is major research area related to XML security [1]. General applications of data integrity could exist in many domains, including e-government, e-commerce, e-financial services, e-business, e-banking, e-healthcare, mobile communications and heterogeneous networks [2-13]. For example, a user contacting a mirror site would need to cryptographically validate the information as genuine, that is, as being the same information as if the response had come directly from the source [13]. Another example has been given by Karl. The MIS department of a company would like to renovate its computer room so as to meet the contemporary hardware requirements. The department needs approval from the Financial Controller and the Estate Department. This requirement first has to be approved by the Financial Controller, and the subsequent approval from the Estate Office will depend on the approval signature of the Financial Controller [2].

Generally, documents pass a hierarchical network of responsibilities (e.g. employees, supervisors) with different roles and access rights. Previous data formats for digital signatures concentrated on signing the entire document, and the XML signature standard is infeasible to make complex workflows secure on a document with multiple signatures. Under this situation, it is necessary to build an XML multi-signature scheme compatible with a natural signing process.

Two XML multisignature schemes have been proposed. The first is based on repeat DSA or RSA scheme. This approach is deployed by W3C in the XML signature specification [14, 15]. It has the drawbacks that the size of a multisignature grows with the increasing of the number of signers and the time for verifying the multisignature is equal to the total time for verifying all personal signature individually.

The second approach is based on a discrete logarithm problem, presented by Lu in 2004. In this scheme, Lu first presented signing rules instead of the message itself. In Lu's scheme, a path expression in XPath is used to transform an XML document into subdocument [16]. For example, let M be the XML data to be cooperatively signed by the signers. XML data M can be divided into set of subdocuments $\{w_1, w_2, \ldots, w_m\}$ by using XPath expression, and then signers only need to sign the XPath instead of message itself. This scheme decreased the communication overhead, although it has three major disadvantages. Firstly, by division, $M = \{w_1, w_2, \ldots, w_m\}$, the integrity checking for each subdocument depends on the formula $h(M) = h(w_1 \| w_2 \| \ldots \| w_m)$. This means the document must be delegated entirely, otherwise the integrity checking will be invalid. For example, a document consists of five parts, and the signers only need to sign three of them. The other two parts have not been

Manuscript received February 5, 2009
Manuscript revised February 20, 2009

delegated, thus the integrity check will be failed. Secondly, the subdocument integrity check needs the signers to check cooperatively online. When the group of signers is small, this is possible, but it is impractical when the group of signers is very large. Thirdly, the scheme only provides broadcast (parallel) signature-generation scenarios. It can not satisfy the natural signing process under a dependent multisignature situation. For example, the company policy is set up in a way that the sequence of approval is important and has to be respected: before launching a project, the financial department has to approve the project. Lu's scheme can not deal with the problem under this application scenario.

Motivated by the problem above, this paper presents a series-parallel XML multisignature scheme. In this scheme, signers are divided into series or parallel subgroups and the members in the signer group can be flexibly managed. The signing order is generated before the signature without a relationship to multisignature scheme. This scheme uses XPath expression to transform XML data, and generates an XML data integrity-checking pool to provide integrity-checking for decomposed XML data. With an integrity checking pool, a signer can check integrity without cooperation from other signers. XML data does not need to be delegated entirely, and signers can complete integrity verification off-line. The series-parallel XML multisignature scheme presented is a mixed-signing order including both series and parallel. If there is a single signer, the scheme is compatible with single XML signature. When each subgroup has a single signer, the scheme is compatible with a series multisignature scheme. When all signers in the same subgroup, the scheme is compatible with a broadcast multisignature scheme. Thus, the scheme presented is closer to a natural application process in practice. Testing shows that the scheme presented has a higher efficiency than repeated DSA or RSA.

1.1 Contribution

The major contribution of the paper presented is an XML multisignature scheme considering a natural signing process, making signing rules more practicable. The detail is as follows.

- The paper presents a series-parallel XML multisignature scheme according to a natural signing process. This scheme is compatible with single XML signatures, sequential and broadcast multisignature schemes.
- The paper presents an XML data integrity-checking pool to provide integrity-checking for decomposed XML data. Only this approach makes signing rules practicable.

1.2 Structure of the paper

The remainder of this paper is organized in sections. Section 2 describes the related work of multisignature scheme and section 3 introduces theory guidance of this research and review of Lu's XML multisignature scheme. Section 4 describes series-parallel XML multisignature scheme, and section 5 presents the experimental results. Section 6 discusses and analyzes efficiency of the scheme presented, and section 7 concludes the paper.

2 Related works

This section introduces the existing multisignature schemes of native approach: extended DSA, RSA, or ElGamal schemes, signing sequence, broadcast signing architecture, distinguished signing authorities, order specify, and XML multisignature schemes.

A native approach, widely used to construct a multisignature for a document, is to repeat the scheme of DSA, RSA, or ElGamal. Such an approach has the drawbacks that the size of a multisignature grows with the increasing of the number of signers and the time for verifying the multisignature is equal to the total time for verifying all personal signature individually.

In order to overcome the drawbacks mentioned above, Italura and Nakamura first proposed a multisignature scheme based on the extended RSA scheme [17]. In this scheme, the size of a multisignature is independent of the number of signers. However, the signers should follow the predefined signing sequence to sign the document, and verify the signature with the knowledge of signing sequence. Similar schemes also can be found in [18, 19, 20, 21], which are based on extended RSA, DSA, or ElGamal schemes with sequential multisignature.

Later, based on a modified ElGamal digital signature scheme, Harn proposed the first multisignature scheme in which the signature-generation and verification does not have to be restricted to the signing sequence [22, 23]. This scheme is known as multisignature scheme based on broadcast architecture, similar to that found in [24, 25]. However, in the schemes mentioned above, all signers sign the same message, and Harn called these schemes multisignature schemes with undistinguished signing authorities. In other words, all signers hold the same responsibility for signing the document. In fact, some applications need to use multisignatures with distinguished signing authorities. For example, a company releases a document that may involve the financial department and

engineering department signing a particular section of the document.

Harn first presented a multisignature scheme with distinguished signing authorities in 1999 [26]. In this scheme, signers can only sign the message for which he or she is responsible. However, Li discovered an efficient insider attack on Harn multisignature scheme with distinguished signing authorities in 2000 [27]. Wu presented a delegated multisignature scheme with document decomposition in 2001 [28]. Wu's scheme is more efficient in multisignature-generation and verification. However, Wu's balanced strategy to delegate subdocuments to qualified signers is problematic, because each signer should sign the portions of the document that they are responsible for rather than the portions of the documents based on some balanced strategy. Mitomi proposed a general model for multisignature with message flexibility in 2000 [29]. Yamamoto improved Mitomi's scheme in 2007 [30]. Wu proposed an ID-based multisignature scheme with distinguished signing authorities for sequential and broadcasting architectures in 2002 [31]. Huang presented multisignatures with distinguished signing authorities for sequential and broadcasting architectures in 2005 [32]. Although these models considered message flexibility, they have not considered the signing order in a natural way.

To date, signing order specified multisignature schemes are Doi's model in 2000, Tada's model in 2002, Burmester's model in 2004, Wang's model in 2005, and Yang's model in 2006 [33, 34, 35, 36, 37]. There are two different major approaches to dealing with this directed series-parallel signing graph. Tada and Yang adopt a series-parallel group defined by [34, 37], which are directed graphs with some characteristic properties. Another approach presented by Burmester, who also represents the group of signers by a graph, and then decomposes the graph to a tree [35]. There are two obvious disadvantages in these schemes. Firstly, the scheme makes the signer order as a signature parameter, increasing the complexity of multisignature algorithm. Secondly, each signer needs to verify the signing order before signing, and update the signing graph or decomposition tree after signing. These disadvantages will lead to inflexibility in adding or deleting signer group members.

As for the XML multisignature scheme, two schemes have been presented. The first is based on a repeat DSA or RSA scheme. This approach is deployed by W3C in XML signature specification [15]. This method has the drawback as mentioned in the native approach for a multisignature scheme. Based on Wu's delegated multisignature scheme, Lu presented XML multisignature in 2004 [16]. In this scheme, he first presented signing rules instead of the message itself. But the integrity check approach will lead to failed integrity verification for decomposed XML data. Furthermore, it is only satisfies the broadcast multisignature application scenario.

3 Theory guidance

3.1 Types of data authentication mechanisms

There are two mechanisms to ensure data authentication as follows:

- Message authentication code
  MAC, a cryptographic check value, is used to provide data origin authentication and data integrity [38]. Both data integrity and data origin authentication can only be provided for the receiving entity. A third party cannot verify these properties, as both sender and receiver are capable to create the MAC (or HMAC).
- Digital signature
  Data is appended, or a cryptographic transformation of a data unit allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery, e.g. by the recipient [39]. More specifically the use of asymmetric encryption provides a means to assure the authentication, also known as non-repudiation.

In this paper, data authentication is ensured by using digital signature. The reasons for adopting digital signature as the data authentication method are used to support requirements for non-repudiation. This is because access to the private key is usually restricted to the owner of the key, which makes it easier to verify proof of ownership. This is the reason for adopting digital signature as data authentication mechanism in this paper.

3.2 Review of Lu's scheme

The scheme in this paper is based on Lu's XML multisignature scheme. This subsection briefly introduces Lu's scheme. In Lu's scheme, there are four components: a group of signer G, a system authority (SA), document decomposition (DD), and a signature collector (SC). DD decompose a document M into a set of subdocuments $\Gamma = \{w_1, w_2, \ldots, w_m\}$ by using a set of rules $T = \{t_1, t_2, \ldots, t_m\}$. Via XPath expression t_i one can easily obtain a subdocument M_i. The procedure for generating a multi-signature of M for G is as follows.

Step 1: DD sends $\{h(M), M_j, T_j\}$ and $\{h(T), h(M)\}$ to u_j and SC, respectively.

Step 2: All $u_j \in G$ extracts w_i from M_j delegated to them and then cooperatively checks the integrity of M by verifying $h(M) = h(w_1 \| w_2 \| \ldots \| w_m)$ where "‖" is the concatenation symbol.

Step3: Every $u_j \in G$ extracts t_i from T_j, computes $w_i = C_{t_i}(M)$, and verifies whether or not every newly computed w_i is identical to the received w_i. If all w_i are successfully verified, each u_j randomly selects an integer $z_j \in Z_q$, computes both

$r_j = \alpha^{z_j} \bmod p$, and

$R_j = r_j^{h(T_j \| r_j)} \bmod p$, and sends R_j to other participant signers and SC.

Step 4: Each $u_j \in G$ computes both

$$R = \prod_{u_k \in G} R_k \bmod p, \text{ and} \quad (1)$$

$$s_j = (z_j h(T_j \| r_j) R + x_j h(h(M \| R))) \bmod q$$

and sends $\{T_j, r_j, s_j\}$ to SC. (r_j, s_j) is the personal signature of M for u_j.

Step 5: SC checks the integrity of T by extracting t_i from the received T_j and verifying whether or not $h(T) = h(t_1 \| t_2 \| \ldots t_m\}$ holds.

Step 6: To verify (r_j, s_j) for every u_j, SC computes R by Eq.1 and checks whether or not the following equation holds.

$r_j^{h(T_j \| r_j) R} = (\alpha^{s_j})(y_j^{h(h(M) \| R)}) (\bmod p)$.

Step 7: If all personal signatures generated in the previous steps are successfully verified, then SC computes

$$S = \sum_{u_j = G} s_j \bmod q$$

and publishes (R, S) as the multi-signature of M for G.

4 Series-parallel XML multisignature scheme

4.1 Series-parallel signing group

- **Signing order graph**

In order to represent signing orders, among n signers, series-parallel graphs has been deployed, which are directed graphs. In this paper, we only deal with graphs with directed (labelled) edges. For an edge labelled i, the initial vertex of the edge is denoted by I_i, and the terminal vertex is denoted by T_i. Thus a directed graph φ can be denoted as $G_d(L(\varphi) = \{i_1, i_2, \ldots, i_k\}; E(\varphi) = \{E_1, E_2, \ldots, E_m\})$, where each i_j is a label of an edge included in the graph φ, and where each E_j is the vertices in the graph. The signers correspond to the vertices in the graph φ. For example, figure 1 shows the signing order in a natural way.

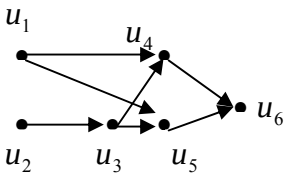


Figure 1 Signing order graph

- **The rules for series-parallel signing group**

Given signers group $SG = \{u_1, u_2, \ldots, u_n\}$, it can be divided into several ordered subgroups according to the following rules.

1. Given signer group SG, it can be defined as $SG = G_1 \cup G_2 \cup \ldots \cup G_n$, and $G_1 \cap G_2 \cap \ldots \cap G_n = \phi$. The signing order is $G_1, G_2, \ldots, G_n$.
2. For $\exists(u_i) \in G_k$, $\exists(u_j) \in G_m$, if $k = m$, then $u_i, u_j \in G_k (= G_m)$, and u_i, u_j can sign parallel. In other words, the signers which are in the same subgroup can sign in parallel.
3. For $\exists(u_i) \in G_k$, $\exists(u_j) \in G_m$, if $k < m$, then u_i, u_j should sign sequentially, and u_i should sign before u_j.

4. For $\exists(u_i) \in G_k$, $\exists(u_j) \in G_m$, if $k > m$, then u_i, u_j should sign sequentially, and u_j should sign before u_i .
5. Only the groups obtained by the rules (1), (2), (3), and (4) are series-parallel signing groups.

- **Sign order graph conversion to series-parallel signing group**

The following algorithm describes how to covert a sign order graph to a series-parallel signing group.

Assume $G = G_1, G_2, \ldots, G_n$ and let $G_k = \phi(k = 1 \ldots n)$.

Step 1: Get labelled edge $i \in \varphi$. The initial vertex is I_i , and the terminal vertex is T_i .

Step 2: If $I_i \notin \exists(G)$, then let $I_i \in G_1$. If $T_i \notin \exists(G)$, then let $T_i \in G_2$.

Step 3: If $I_i \in \exists(G_k)$, and $T_i \notin \exists(G)$, then let $T_i \in G_{k+1}$. Otherwise, assume $T_i \in (G_m)$. If $m \le k$, then move T_i from G_m to G_{m+1} , until $m > k$.

Step 4: Go to step 1 until each edge in φ has been handled.

According to above algorithm, signing order graph figure 1 can be converted to the following series-parallel signing group.

$G_1 = \{u_1, u_2\}, G_2 = \{u_3\}, G_3 = \{u_4, u_5\}, G_4 = \{u_6\}$

This means signers can generate a parallel signature in each subgroup, where every subgroup signing is sequential. Thus, the converted signing order is as shown in figure 2.

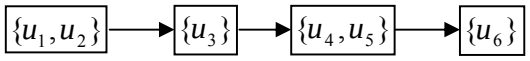


Figure 2 Converted series-parallel signing order

4.2 XML data decomposition (XDD)

XML data has a simple data model based on trees. DOM is a standard interface (API) that defines how XML data are to be accessed. DOM is naturally a tree-like representation; as such, it admits a bottom-up hashing procedure. The full details of DOM-HASH are available in [40]. For our integrity verification purpose, the important properties of DOM-HASH are as follows.

If the root hash of an XML document X_D is known to a signer u_i , it is possible to provide evidence to u_i that any subtree st_i of the XML data occurs under X_D without revealing all of X_D and online verification. First, note that u_i can DOM-HASH the subtree st_i to get the root hash of st_i . Now u_i can be given just the hash value of the siblings of st_i and the sibling of all its parents, and u_i can recompute the root hash of X_D . Since the hash function is assumed to be one-way, signer u_i can be reasonably sure that the hash values could not have been forged, and that st_i really did occur in X_D . The same process can be used to prove that one subtree st_i occurred under another subtree st_j within the same XML data, using the hash values along the XPath from st_i to st_j , without revealing any other subtree under st_j .

It is given an XML data X_D , a DTD conforming to the XML data, and a pool τ with a finite number of entries, for each possible Xpath in DTD. Thus, the integrity checking pool can be defined as follows.

Definition 1 XML data integrity checking pool τ , τ is a tuple as $(p, h(p), c(p), h(c(p)))$, here

- p is the possible XPath in the DTD.
- $h(p)$ is the digest value of each p , and h is a secure one-way hash function.
- $c(p)$ denotes the content according to XPath p ,
- $h(c(p))$ is the digest value of $c(p)$.

The process for XML data integrity checking pool τ generation is as follows.

1. Generate a possible Xpath $\forall p_i \in p, i \in N$ in the DTD, and related digest value $h(p_i)$ .Insert p_i , p_i related content $c(p_i)$, and $h(p_i)$ into pool τ
2. Build DOM-HASH associates a secure hash value $h(c(p_i))$ with each p_i , and let $m_i = h(c(p_i))$.
3. There could be many sub-trees $st_i, i \in N$ associated with each p_i in the Xpath table τ . These are digested together using the concatenation hash function, $m_i = h(st_1 \| st_2 \| \ldots \| st_n)$ to give a digest value each entry p_i .

For integrity verification, there is the pool τ with Xpath entries p_i, an integrity verification request q from a signer.

1. Match q against each entry in τ.
2. If q matches the entry, retrieve the hash value m_i related to the entry p_i. If there is no corresponding entry matched to q, reject, otherwise, go to step 3.
3. Build digest value $m_i^{'}$ with step 3, check that $m_i^{'} \stackrel{?}{=} m_i$. If not reject, otherwise, accept. After accepting, if signer does not believe in this result, the verification process can be extended to parent verification as shown in step 4.

Assume $q^{'}$ is the Xpath of q parent, let $q = q^{'}$, go to step 1. Finally, signer can generate the hash value of the whole XML data X_D, check that $m_i^{'} \stackrel{?}{=} m_i$. If it is not equal, reject, otherwise accept. This is a convincing result, because the integrity of whole XML data has been checked.

4.3 XML multisignature scheme

The system has the following roles: a group of signers, a system authority (SA), an XDD, and a signature collector (SC). The services provided by SA are to initialize system parameters, and to generate the secret keys and public keys for the group and the signer. The services provided by XDD are to decompose the XML data to be signed into a set of sub-data. The services provided by SC are to collect and verify the personal signatures generated by the signers, and to construct a multisignature for the XML data from these verified personal signature. For simplicity, it is assumed that all signers trust SA and SC. The proposed scheme operates through the following three stages: the secret key/public key generation stage, the multisignature generation stage, and the multisignature verification stage.

1. Common parameters
   The common parameters are similar to those defined in [41] for DSA standard to which the group dimension has been added. Assuming a group of n signers, where s_1 is the group manager GM, the following parameters are defined:
   - p, q : Two large prime numbers such that $q \mid (p-1)$ as defined in digital signature algorithm [41].
   - g : Generator of the cyclic group of order q in Z_p^* (selects an element $h \in Z_p^*$ and computes $g = h^{(p-1)/q} \bmod p$ such that $g \neq 1$).
   - $x_1, x_2, \ldots, x_n$: Group members' private keys.
   - $y_1, y_2, \ldots, y_n$: Group members' public keys such that $y_i = g^{x_i} \bmod p$ is computed.
   - SA generates a secret key/public key pair (X_i, Y_i) for each subgroup G_k, where
     $$X_i = \sum_{u_{ji} \in G_k} x_j \bmod q \quad (2)$$
     $$Y_i = \prod_{u_j \in G_k} y_j \bmod p \quad (3)$$
   - $h(.)$: A cryptographic strong hash function (one-way function) such as SHA-1, SHA-2.

2. Signature generation and verification

The procedure for generating a multisignature of X_D for G is described as follows.

Step 1: XDD sends $\{\tau, X_D, T_j\}$ to u_j, and $T_j = \{p_1, p_2, \ldots, p_j\}$

Step 2: Every $u_j \in G$ extracts X_D^j from X_D using T_j, and then checks the integrity of X_D^j by τ and the integrity verification process.

Step 3: If integrity of X_D^j is successfully verified, each $u_j \in G_k, j,k \in [1,N]$ randomly selects an integer $z_j \in Z_q$, computes

$$r_i = g^{z_j} \bmod p, \quad (4)$$

and sends $\{T_j, r_j\}$ to other participant signers in the same subgroup and SC.

Step 4: After receiving $\{T_j, r_j\}$, $u_i (i \neq j)$ and SC can compute $R_j = r_j^{h(T_j \| r_j)} \bmod p$ (5)

Step 5: Each $u_j \in G_k, j,k \in [1,N]$ computes both

$$R_k = \prod_{u_j \in G_k} R_j \bmod p \,(6),$$

$$s_j = (z_i h(T_j \parallel r_i) R_k + x_j h(h(\tau) \parallel R_k)) \bmod q \,(7),$$

and sends $\{s_j\}$ to SC. (r_j, s_j) is the personal signature of X_D by signer u_j.

Step 6: To verify (r_j, s_j) for every $u_j \in G_k, j,k \in [1,N]$, SC computes R_k by Eq. (6) and checks whether or not the following equation holds.

$$r_j^{h(T_j \| r_j) R_k} = (g^{s_j})(y_j^{h(h(\tau)\| R_k)}) \bmod p \quad (8)$$

Step 7: If all personal signatures generated in the previous steps are successfully verified, then SC computes

$$S_k = \sum_{u_j \in G_k} s_j \bmod q \quad (9)$$

and publishes (R_k, S_k) as the multisignature of X_D by subgroup G_k.

Signature verification for subgroup:

For verifying the subgroup multisignature (R_k, S_k), the verifier checks the following equality:

$$R_k^{R_k} = (g^{S_k})(Y^{h(h(\tau)\| R_k)}) (\bmod p) \quad (10)$$

If Eq. (10) holds, then subgroup multisignature (R_k, S_k) is successfully verified.

The signature of the whole group (this signature is used to ensure signing order):

Step 1: SC verifies each subgroup multisignature (R_k, S_k), if any of them are invalid, then reject, otherwise, go to step 2.

Step 2: SC computes $S_G = h(S_1 \| S_2 \| \ldots \| S_k)$, here $S_i, i \in [1..k]$ is each subgroup signature.

Step 3: The signature for subgroup G_1:

$$\sigma_1 = g^{k_1} \bmod p \quad (11)$$

$$\rho_1 = S_G X_1 - \sigma_1 k_1 \bmod q \quad (12),$$

and sends (σ_1, ρ_1) to next subgroup.

Step 4: For subgroup G_i, first verifying the signature by G_{i-1} through

$$g^{\rho_{i-1}} \prod_{j=1}^{i-1} \sigma_j^{\sigma_j} = \prod_{j=1}^{i-1} Y_i^{SG} \bmod p \quad (13)$$

If this generates a failed verification, then reject the signature from G_{i-1}, otherwise, compute

$$\sigma_i = g^{k_i} \bmod p, \quad (14)$$

$$\rho_i = \rho_{i-1} + S_G X_i - \sigma_i k_i \bmod q \quad (15)$$

Then (σ_i, ρ_i) is the final multisignature for group SG.

Step 5: Verification for final multisignature:

$$g^{\rho_i} \prod_{j=1}^{k} \sigma_j^{\sigma_j} = \prod_{j=1}^{k} Y_i^{S_G} \bmod p \quad (16)$$

4.3 Correctness proofs

Since proposed scheme for subgroup signature is based on Lu's scheme. Thus, correctness of the single signature and subgroup signature is as their scheme. Here, this paper just provides the proofs of sequential signature for subgroup.

Theorem 1 If equation (13) is true, then the subgroup signature (σ_i, ρ_i) is valid

Proofs: From (15), for $\exists(i)$,

$$\sum_{j=1}^{i} \sigma_j k_j + \rho_i = \sum_{j=1}^{i} \sigma_j k_j + \rho_{i-1} + S_g X_i - r_i k_i \bmod q$$

$$= \sum_{j=1}^{i-1} \sigma_j k_j + \rho_{i-2} + S_G (X_i + X_{i-1}) - \sigma_{i-1} k_{i-1} \bmod q$$

$$= \sum_{j=1}^{i} S_G X_i \bmod q$$

Then, $g^{\sum_{j=1}^{i} \sigma_j k_j + \rho_j \bmod q} = g^{\sum_{j=1}^{i} S_G Y_j \bmod q} \bmod p$

$$= \prod_{j=1}^{i} (g^{X_i})^{S_G} \bmod p$$

$$= \prod_{j=1}^{i} (Y_i)^{S_G} \bmod p$$

Thus, the Eq. (13) is correct.

Theorem 2 If equation (16) is true, then the final signature for group is valid.

Proofs: Because Eq. (16) is a special expression from Eq. (13), for $i = k$, then Eq. (13) is equal to Eq. (16). Thus, the Eq. (16) is correct, and the sequential signature for group is valid.

4.4 Security analysis

The security of the proposed scheme is based on the following well-known cryptographic problems, which are also frequently used as the basics for analyzing the security strength of the contemporary crypto-schemes or algorithms. The security of the proposed scheme is as secure as Wu's scheme which is based on the discrete logarithm and one-way hash function problems. Note that there are two particular issues that need to be addressed.

The security problem related to presented scheme is as follows:

- Issue 1: Forging an integrity verification table τ
  Assume (given an XML data and a conforming DTD) that the decomposition process is executed correctly. Now the signer will always reject an incorrect answer and accept a correct one, unless a collision in the hash function used in decomposition process.

  Analysis of issue 1: Assume here that the signer uses the DTD to compute the precise set of table entries which matches his XML data to be signed. The argument that the signer will accept correct XML data is straightforward, based on the fact that he simply repeats the computation done by the decomposition process and results in the same digest value. Now we argue that the signer will reject any incorrect XML data to be signed. It is sufficient to establish that the signer will not accept the wrong set of subtrees from any table entry. If an adversary sends an incorrect subtree, then the corresponding DOM-HASH will be different from that used in the process of computing the digest for that table entry. So the adversary has to have found a second pre-image that hashes to the same value in some steps in the process of computing the digest for an entry; alternatively the adversary has to have found a hash collision in some steps of the process of computing the digest for the entire table. In either case, the publisher has to engineer collisions in the hash functions that produce a specific output. For a secure one-way hash function h, given $y = h(x)$, it is computationally unfeasible to find $x_1 \neq x_2$, such that $h(x_1) = h(x_2)$

- Issue 2: Forging a multi-signature

The signature generated by the last subgroup is the multisignature (S_G, σ_i, ρ_i), the verification equation is Eq. (16). The security of Eq. (16) is expressed by theorem 3.

Theorem 3 It is a DLP problem to calculate ρ_i through (S_G, σ_i), or to calculate σ_i through (S_G, ρ_i) in Eq. (16).

Proofs: From Eq. (16), it is easy to understand that it is a DLP problem to calculate ρ_i via (S_G, σ_i).

Given (S_G, ρ_i), then g^{ρ_i} and $\prod_{j=1}^{i} Y_j^{S_G}$ are constants.

Let $C_1 = g^{\rho_i}$, $C_2 = \prod_{j=1}^{i} Y_j^{S_G}$, then Eq. (16) can be rewritten as: $\sigma^{\sigma} C_1 = C_2^{\sigma} \bmod p$, then has $(\sigma C_2^{-1})^{\sigma} = C_1^{-1} \bmod p$ (17)

We can get $C_3 = C_1^{-1}$, and $C_4 = C_2^{-1}$ in $GF(p)$. Then Eq. (17) can be written as:

$(\sigma C_4)^{\sigma} = C_3 \bmod p$, thus,

$(\sigma C_4)^{\sigma C_4} = (C_3)^{C_4} \bmod p$ (18),

Assume $\sigma C_4 = X$, and $(C_3)^{C_4} = C$, then Eq. (18) can be written as:

$X^X = C \bmod p$ (19)

Thus, given (S_G, ρ_i) to calculate σ_i is equal to get X from Eq. (19). It is easy to identify it is a DLP problem to get X from Eq. (19).

5 Performance evaluation

5.1 Evaluation environment

All the testing was performed on a PC with a 2.39 GHz Pentium (R) 4 processor, 0.99GB of RAM, and the MS Windows XP operating system. The algorithms have been coded in C#.net. The evaluation has taken into account two parameters: the number of signers, and the number of bits used to generate the common parameters.

5.2 Evaluation results

Figures 3 and 4 show the execution time overhead corresponding to the signing process, while figures 5 and 6 show the execution time overhead corresponding to the verifying process.

Figure 3 and 4 show that the superiority of the scheme presented in this paper and Lu's scheme over RDSA increasing with the signers size. Although all signers should sign specific XML data, the scheme in this paper and scheme by Lu have almost 50% higher efficiency. The major reason for this result is that these two schemes only sign the rules, instead of the XML data itself. Compared to sign XML data itself, the rules are significantly smaller. This will decrease the time taken to generate the digital

value. Compared to Lu's scheme, the two have almost the same efficiency, however, the scheme presented has more functionality and is more practicable in applications.

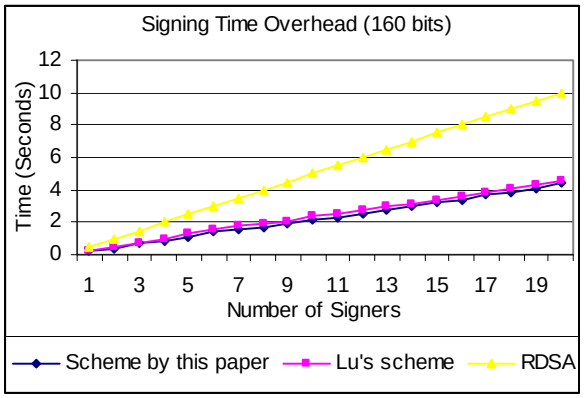


Figure 3 Execution time comparison (160 bits signing)

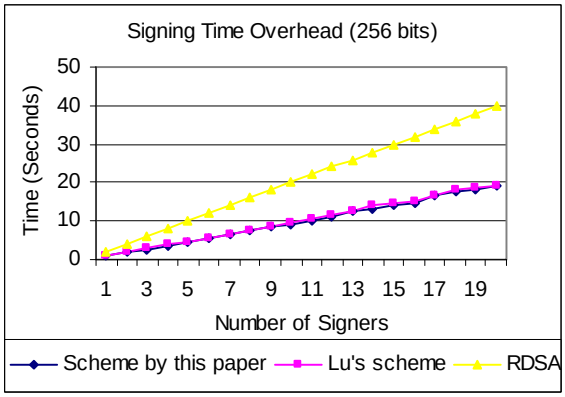


Figure 4 Execution time comparison (256 bits signing)

Figure 5 and 6 show the superiority of scheme presented in this paper and scheme presented by Lu over RDSA in terms of execution times. The figures show that increasing the size group has less impact on schemes both in this paper and by Lu. When a signature is verified, RDSA should check each signature generated by signers, and this leads to a line of increasing verification time. But the schemes presented both in this paper and by Lu only need to verify the signature generated by SC, thus, the verification time almost is a constant of about 1.2 seconds.

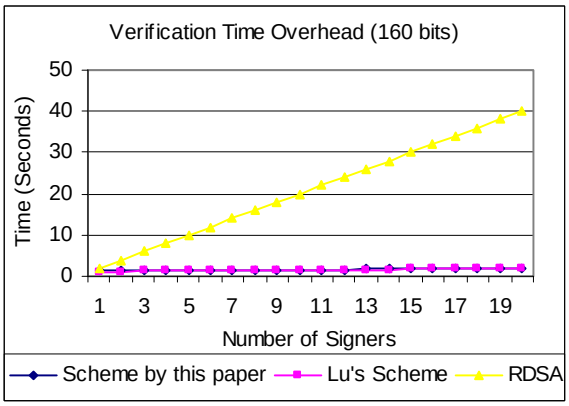


Figure 5 Execution time comparison (160 bits verification)

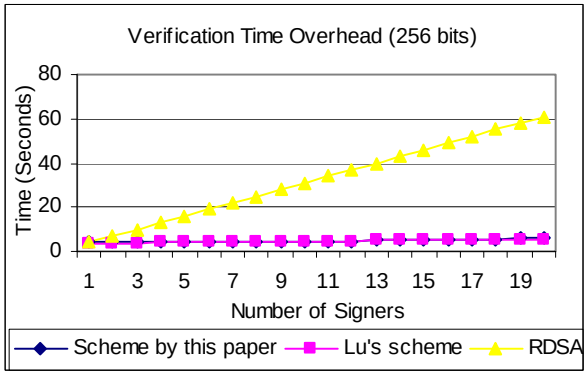


Figure 6 Execution time comparison (256 bits verification)

6 Discussion & Analysis

6.1 Efficiency analysis

The performance of the proposed scheme depends on the requirement of time complexity (measured by the multisignature generation stage and the multisignature verification stage, respectively). Let T_m, T_e, and T_h be the time required to perform a modular multiplication, a modular exponential, and the one-way hash function h; respectively. The following symbols are used for evaluating the performance of the proposed scheme: n is the number of signers in G; k is the number of divided subgroup for G; and i is the signer's number in subgroup G_k.

The time complexities for generating and verifying a personal signature (r_i, s_i) are identical to Lu's scheme; the

time complexities of both stages are $O((n+2)T_m + 2T_e + 3T_h)$ and $O(T_m + 3T_e + 2T_h)$; respectively. The time complexities for generating and verifying a subgroup signature are different from the signers in the subgroup, both stages are $O((i-1)T_m + iT_e + (i+2)T_h)$ and $O(T_m + 3T_e + 2T_h)$; respectively. The worst situation is where all the signers are in the same group, that is $i = n$. The time complexities for construct multisignature from subgroup are $O((k+2)T_m + 3T_e + T_h)$ and $O(T_m + 3T_e + T_h)$.

The time complexities for integrity verification table τ consist of the following two factors: the node size and the depth size. In a $k-ary$ tree with a depth of m, in the worst situation, then number of nodes that could be hashed is $N = \sum_{x=1}^{m} k^{x-1} = \frac{k^m - 1}{k-1}$, and the number of hash required $W = \sum_{x=1}^{m} xk^{x-1} = \frac{mk^{k+1} - (m+1)k^m + 1}{(k-1)^2}$.

The time complexity of an iterative hash function h can be described as a function of its input size l by the function, $T(l) = c_1(\left\lfloor \frac{l}{D} \right\rfloor + 1) + c_2$, where D is constant [42]. If v is a vertex of XML data X_D, $in\deg(v)$ denotes the depth of vertex v, that is the number of predecessors of v in X_D. Let S be a subtree of X_D. The two components of the integrity cost for S are defined as follows. The node size S_n of S is the number of its vertices. The depth size S_d of S is the sum of the depth of its vertices, that is $S_d = \sum_{v \in S} in\deg(v)$. Then, the rehashing overhead is given by a linear combination of the node size and the depth size of S, that is $c|v| + c' \sum_{v \in S} in\deg(v) = cS_n + c'S_d$, where both c and c' are constants. The verification time is a quantity of the form $c|v| + c' \sum_{v \in S} in\deg(v)$.

6.2 Compatibility with XML Signature Specification

The "XML signature Syntax and Processing" recommendation is an internet standard which defines a syntax and processing model of a special format for digital signatures. XML signature based on XML technology was standardized in February, 2002, through the efforts of W3C and IETF [15]. Standardized contents describe clear statement of the regulations on XML signature to maximize the security and the extent of the standardized contents, integrity, message and user authentication and non-repudiation. These signatures are represented in an XML format and can sign arbitrary resources, including XML and parts thereof.

The structure and processing of XML signatures introduces some interesting concepts which will be explained briefly. The primary elements of XML signatures are digital signature information and digest value information (The presentation of XML schema is as shown in figure 7). Signature elements consist of "SignedInfo" with digital signature information, "SignatureValue" with actual digital signature value and "KeyInfo" with digital signature key information. In particular, "SignedInfo" describes how signature information is standardized, the algorithm for the signature and the subordinate algorithm. "Reference" consists "DigestMethod", the algorithm summarizing signature data, and the element "DigestValue" showing the result. "KeyInfo" described in XML security is used to illustrate key information in XML digital signature.

```
<Signature>
 <SignedInfo>
   <CanonicalizationMethod/>
   <SignatureMethod/>
   <Reference>
    <DigestMethod/>
    <DigestValue/>
  </Reference>
 </SignedInfo>
 <SignatureValue>
 <KeyInfo>
<Signature>
```

Figure 7 XML digital signature elements

As described in our proposed scheme, each signer $u_i \in G$ extracts rules p_i from the set of rules T delegated to him. Therefore, we can use "Transforms" elements to describe p_i 's content need to be signed. Other information can also be defined in an XML Signature. For example, the hashing function h can be described in the "DigestMethod" element and the signature value can be written into the "SignatureValue". Therefore, the proposed XML multisignature scheme is compatible with the XML Signature standard.

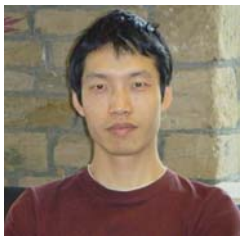
7 Conclusion & future work

XML data authentication is a major research area related to XML security, and it has a wide range of applications in practice. This paper presents a series-parallel XML multisignature scheme. In the scheme presented, the signer group is divided into series or parallel subgroups. The scheme uses XPath expression to transform XML data, and generates an XML data integrity checking pool to provide integrity checking for decomposed XML data. Through testing, the scheme is compatible with XML signature specification. The performance evaluation shows that the scheme presented has a higher efficiency than repeated DSA or RSA. The new scheme can be used directly in many applications, for example, in e-business for a joint signature of a contract between two or more organizations, or in e-government to sign an electronic document with different department under different roles.

References

[1] E.Bertino. XML security. Information security technical report, vol 6.no2(2001) 44-58.
[2] Karl R.P.H. Leung, Lucas C.K. Hui, Handling signature purposes in workflow systems, The Journal of System and Software, 55 (2001) 245-259.
[3] C. Wu, H. Shan, W. Wang, D. Shieh, M. Chang, E-Government Electronic Certification Services in Taiwan, proceedings of the Second International Workshop for Asian Public Key Infrastructures, (Taipei, Taiwan, 2002) 1-8.
[4] Y.H. Chen, E.J. Lu, Design of a secure fine-grained official document exchange model for e-government, Information & Security 15(1) (2004) 55-71.
[5] A. Rushinek, S. Rushinek, E-commerce security measures: are they worth it?, Ubiquity 3(39) (2002) 1.
[6] J.E. Boritz, W.G. No, Security in XML-based financial reporting services on the Internet, Journal of Accounting and Public Policy 24(1) (2005) 11-35.
[7] S. Jones, M. Wilikens, P. Morris, M. Masera, Trust requirement in e-business, Communications of the ACM, 43(12) (2000) 81-87.
[8] M. O'Neill, Case Notes from a Vulnerability Assessment of a Bank's Web Services, XML2007 conference & Exposition (Massachusetts, USA, 2007) 18-24.
[9] B. Blobel, Authorisation and access control for electronic health record systems, International Journal of Medical Informatics 73(3) (2004) 251-257.
[10] J. Dankers, T. Garefalakis, R. Schaffelhofer, T. Wright, Public key infrastructure in mobile system, Electronics and Communication Engineering Journal 14(5) (2002) 180-190.
[11] S. Karnouskos, Security-enabled code deployment for heterogeneous networks, Computer Standards & Interfaces 27(5) (2005) 547-560.
[12] Ekelhart, A., Fenz, S., Goluch, G., Steinkellner, M., Weippl, E., 2007. XML security – A comparative literature review. Journal of Systems and Software 81 (2008), pp. 1715-1724.
[13] E.Damiani, S.De, C.Di, P.Samarati. Towards securing XML web services. ACM workshop on XML security. November 22, 2002, USA.
[14] Wolfgang Kubbilun, Sebastian Gajek, Michael Psarros and Jörg Schwenk. Trustworthy Verification and Visualisation of Multiple XML-Signatures. LNCS. Volume 3677/2005. pp311-320.
[15] M. Bartel, XML Signature Syntax and Processing. Available at: http://www.w3.org/TR/xmldsig-core/
[16] Eric Jui-Lin Lu, Rai-Fu Chen, An XML multisignature scheme, Applied Mathematics and Computation, 149 (2004) 1-14.
[17] Itakura, K., Nakamura, K., 1983. A public-key cryptosystem suitable for digital multisignatures. NEC Research and Development 71, pp 1-8.
[18] Harn, L., Kiesler, T., 1989. New scheme for digital multisignature. Electronics letters 25, pp 1002-1003.
[19] Kiesler, T., Harn, L., 1990. RSA blocking and multisignature schemes with no bit expansion. Electronics letters 26, pp.1490-1491
[20] Ohta, K., Okamoto, T., 1991. A digital multisignature scheme based on the fiat-shamir scheme. Advances in Cryptology, ASIA Crypt'91, Springer, Berlin, pp. 139-148
[21] Boyd, C., 1991. Multisignatures based on zero knowledge schemes. Electronics letters 27, pp 2002-2004.
[22] Harn, L., 1994a. Group-oriented (t,n) threshold digital signature scheme and digital multisignature. IEE Proceedings Computers and Digital Techniques 141, 307-313
[23] Harn, L., 1994b. New digital signature scheme based on discrete logarithms. Electronics letters 30, 396-398.
[24] Hardjono, T., Zheng, Y., 1992. A practical digital multisignature scheme based on discrete logarithms. Advances in Cryptology, AUSCRYPT'92, Springer, Berlin, pp. 122-132
[25] Michels, M., Horster, P., 1996. On the risk of disruption in several multiparty signature shcemes. Advances in Cryptology, ASIA Crypt'96, Springer, Berlin, pp. 125-132
[26] L. Harn. Digital multisignature with distinguished signing authorities. Electronics Letters, Volume 35, Issue 4, 18 Feb 1999 pp. 294 – 295.
[27] Z.C.Li, L.C.K. Hui, K.P. Chow, C.F. Chong, W.W. Tsang amd H.W. Chan, Cryptanalysis of Harn digital multisiganture scheme with distinguished signing authorities. Electronics Letters, Volume 36, Issue 4, 17 Feb 2000 pp. 314 – 315.
[28] Tzong-Chen Wu, Chih-Chan Huang, D. -J. Guan, Delegated multisignature scheme with document decomposition, The Journal of Systems and Software 55 (2001) 321-328.
[29] S. Mitomi, Atsuko Miyaji: A general model of Multisignature Scheme with Message Flexibility, Order Flexibility and Order Verifiability. ACISP 2000: pp. 298-312.
[30] Dan Yamamoto, Wakaha Ogata, A General Model of Structured Multisignatures with Message Flexibility, IEICE Trans. Fundamentals, Vol. E90-A, No. 1 January 2007. Pages 83-90
[31] Tzong-Sun Wu, Chien-Lung Hsu. ID-based multisignatures with distinguished signing authorities for sequential and broadcasting architectures. Applied Mathematics and

Computation, Volume 131, Issues 2-3, 25 September 2002, Pages 349-356
[32] Hui-Feng Huang, Chin-Chen Chang, Multisignatures with distinguished signing authorities for sequential and broadcasting architectures. Computer Standards & Interfaces, Volume 27, Issue 2, January 2005, Pages 169-176.
[33] Hiroshi Doi, Masahiro Mambo, and Eiji Okamoto. On the security of the RSA-based multisignature scheme for various group structures. ACISP 2000, LNCS 1841, pp. 352-367, 2000.
[34] Mitsuru Tada, An Order-Specified Multisignature Scheme Secure against Active Insider Attacks, ACISP 2002, LNCS 2384, pp.328-345.
[35] Mike Burmester, Yvo Desmedt, Hiroshi Doi, and Masahiro Mambo. A structured ELGamal-Type Multisignature Scheme. Lecture Notes in Computer Science, Volume 1751/2004, pp. 466-483.
[36] L. Wang, E. Okamoto, Y. Miao, T. Okamoto, and H. Doi, "ID-based series-parallel multi signature scheme for multi-message from bilinear maps," International Workshop on Coding and Cryptograph (WCC2005), LNCS 3969, pp.291–303, Springer-Verlag, Berlin, 2006.
[37] M. Yang, L. Su, J. Li, and F. Hong. Secure order-specified multisignature scheme based on DSA. Wuhan University Journal of Natural Science. Vol. 11 No. 6 2006 pp. 1614-1616
[38] Information technology -- Open Systems Interconnection -- Security frameworks for open systems: Non-repudiation framework
[39] Information processing systems -- Open Systems Interconnection -- Basic Reference Model -- Part 2: Security Architecture
[40] H. Maruyama, K. Tamura, N. Uramoto, Digest Values for DOM (DOMHASH), RFC2803, http://www.landfield.com/rfcs/rfc2803.html, April 2000.
[41] National Institute of Standard and Technology, Digital Signature Standard, FIPS Publication,186-3, 2006.
[42] R. Tamassia, N. Triandopoulos, On the Cost of Authenticated Data Structures. In Proc. European Symposium on Algorithms, LNCS (2832) (Budapest, Hungary, 2003).



Baolong Liu received B.S. and M.S. degrees in Computer Science and Engineering from Shaanxi University of Science & Technology in 1998 and 2003, respectively. Now he is a doctoral candidate in the School of Computing & Engineering, University of Huddersfield. His research interest is in XML data integrity, authentication, and confidentiality.